Deep Features or Not: Temperature and Time Prediction in Outdoor Scenes

Anna Volokitin
Computer Vision Laboratory
D-ITET, ETH Zurich
voanna@vision.ee.ethz.ch

Radu Timofte
Computer Vision Laboratory
D-ITET, ETH Zurich
timofter@vision.ee.ethz.ch

Luc Van Gool
CVL, D-ITET, ETH Zurich
PSI, ESAT, KU Leuven
vangool@vision.ee.ethz.ch

Abstract

In this paper, we study the effectiveness of features from Convolutional Neural Networks (CNN) for predicting the ambient temperature as well as the time of the year in an outdoor scene. We follow the benchmark provided by Glasner et al. *[3] one of whose findings was that simple hand-crafted features are better than the deep features (from fully connected layers) for temperature prediction. As in their work, we use the VGG-16 architecture for our CNNs, pre-trained for classification on ImageNet. Our main findings on the temperature prediction task are as follows. (i) The pooling layers provide better features than the fully connected layers. (ii) The quality of the features improves little with fine-tuning of the CNN on training data. (iii) Our best setup significantly improves over the results from Glasner* et al. *showing that the deep features are successful in turning a camera into a crude temperature sensor. Moreover, we validate our findings also for time prediction and achieve accurate season, month, week, time of the day, and hour prediction.*

1. Introduction

The perception of the world set deep roots in humans' logos. A color, beyond its physical nature and wave length, can be 'beautiful' and 'warm' or 'neutral' and 'cold'. The subjective interpretation of a physical property is due to the way the humans respond to their habitat, to the nature, and to life events. For the temperate climate, white correlates with 'cold' and 'winter' due to the winter's snow, green to 'fresh' and 'summer' as the flora turns green in 'spring' and 'summer', and bright yellowish light connects to 'warm' and 'summer' due to the summer sunlight. These correlations besides human senses and subjective interpretations are factual – snow is unlikely to fall in summer. Given an outdoor scene one can have an educated guess on the ambient temperature only by looking at the amount of light and the saturation of the colors in relation to the natural and/or man-made objects. The same can be said about the time of the year or the hour of the day.

+17°C, spring, month 04, week 15, day 103 +6°C, fall, month 11, week 46, day 315

−4°C, winter, month 12, week 49, day 332 +23°C, summer, month 7 week 27, day 194

Figure 1. Appearance varies with temperature and time of the year.

In this paper we focus on the prediction of both the ambient temperature and the time of the year at level of season, month, week, or even day, from an image of an outdoor scene (see Fig. 1). Also, we predict the hour and the time of the day.

The correlations between appearance and temperature in outdoor scenes were explored by the very recent work of Glasner *et al*. [3]. As a change in temperature of an object or substance usually alters its appearance (*e.g*. iron at high temperatures turns red, glows, and melts; thermometers are based on the dilation effect), one should be able to find correlations between the outdoor scene appearance and ambient temperature. Glasner *et al*. study these correlations to finally achieve impressive temperature prediction results for images depicting outdoor scenes. The features and the models Glasner *et al*. found to be most reliable for temperature prediction are generally hand-crafted, and this in contrast with the convolutional neural networks (CNN) features which are learned. We, on the other hand, consider that the representations learned through CNN training on a specific task should be more powerful than the handcrafted ones especially when sufficient training data is available and the relation between the inputs and outputs is not very well understood. In the recent years CNNs and the deep features

978-1-5090-1437-8/16 $31.00 © 2016 IEEE
DOI 10.1109/CVPRW.2016.145

extracted from CNNs were applied to a large number of vision tasks and led to state-of-the-art results (*e.g.* image classification [15]). Therefore, we continue the work of Glasner *et al*. [3] and focus on the deep CNN features using the VGG16 [15] architecture. As shown by our experiments, for temperature prediction the pooling layers make better features that the fully connected layers and lead to substantially better performance than the results of Glasner *et al*. (a $0.7°$C average error reduction).

The prediction of the time of the year, under the form of season, month, week, or even day prediction is the other focus and a novelty of our work. As in the case of ambient temperature prediction, we work under the assumptions that appearance strongly correlates with the temperature and time of the year and use the same deep features for both tasks. The task is more difficult than the temperature prediction as the time of the year is not directly predicted by the temperature but is the result of multiple interactions between the scene objects, the temperature, and sunlight. As is the case for humans, the accuracy of our prediction is poor at day level ($\sim 1\%$), gets significantly better at week-level ($\sim 10\%$), to then reaches good performance at month-level ($\sim 50\%$ or 2.4 months average error) and at season-level ($\sim 69\%$ or 0.6 season average error).

Our main contributions are:

1. An analysis of (deep) features for the task of temperature prediction in outdoor scenes.
2. A novel time prediction task in outdoor scenes.
3. A dataset of time-lapse sequences and their corresponding day, week, month, and season of the year annotations.
4. Large improvements on temperature prediction and robust accuracy on time prediction for outdoor scenes.

1.1. Related Work

As early as 1998, Szummer and Picard [16] addressed the problem of indoor-outdoor image classification and studied several low level image features.

In a series of papers Narasimhan, Nayar, and their coauthors (*e.g.* [12, 13]) study the outdoor images in relation to weather. They start from a physics foundation and build models that capture the weather effect on the images. Also, they introduce in [13] the WILD dataset with calibrated and registered images of a fixed outdoor scene exhibiting a wide range of weather conditions.

The largest public database with outdoor webcam images is the Archive of Many Outdoor Scenes (AMOS), a project started in 2007 by Jacobs, Pless, and their collaborators [7]. By now, AMOS collected 884 millions of images and still counting from publicly accessible outdoor webcams from all over the world and therefore has a broad diversity of contents. Jacobs *et al*. [6, 5] use AMOS to predict wind velocity and vapor pressure while others such as Islam *et al*. [4] align collected weather data to webcams from AMOS and study the utility of such information in predicting scene appearance.

Laffont *et al*. [9] study what they call "transient attributes"(TA) and their effect on the outdoor scene appearance. TA are high level properties such as "spring", "rain", and "fog". For each such TA they train regressors for estimation and demonstrate the synthesizability of the appearance of a scene for different weather conditions with an image editing application.

Recently, Lu *et al*. [10] propose a collaborative learning approach for labeling outdoor images as either sunny or cloudy and a corresponding annotated dataset with 10,000 images. Under the same settings, Elhoseiny *et al*. [2] use convolutional neural networks for classification. Murdock *et al*. [11] go further and connect webcam observations to satellite imagery so that to build cloud maps directly from ground level webcam readings across USA.

The most related work to ours is the very recent work of Glasner *et al*. [3]. They study the ambient temperature prediction starting from an image of a specific outdoor camera with known past recordings. They succeed to achieve impressive prediction with handcrafted features and simple regression models. We, on the other hand, analyze and support the use of deep features for this task and consistently improve upon their results.

For the first time, to the best of our knowledge, we address prediction of the time of the year from an outdoor input image for a specific camera, and this with a good accuracy.

The remainder of the paper is structured as follows. First, in Section 2 we describe the experimental setup of our study, then, in Section 2.3, we analyze the deep features with and without fine-tuning on training data and compare their performance for temperature prediction. In Section 3 we explore a related task, the time prediction in outdoor scenes and validate the same deep features by achieving good results. We conclude the paper in Section 4. Code is available at https://github.com/voanna/Deep-Features-or-Not

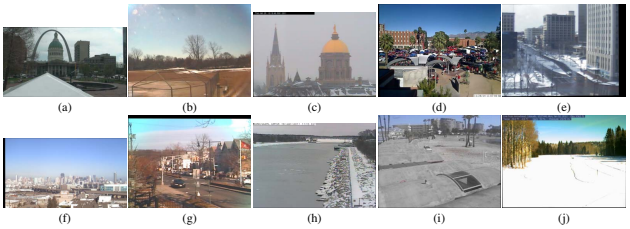

Figure 2. Sample images from Glasner *et al*. GB dataset [3] for each sequence (a–j).

GD[3]	(a)	(b)	(c)	(d)	(e)	(f)	(g)	(h)	(i)	(j)
train	287	298	334	321	320	234	329	341	291	313
test	257	321	348	340	333	230	346	354	276	299
TYD (ours)	(i)	(ii)	(iii)	(iv)	(v)	(vi)	(i-vi)			
train	2048	5357	2348	2735	2012	2475	16975			
test	86	5308	235	208	310	76	6223			

Table 1. Datasets for temperature prediction (GD) [3] and time of the year prediction (TYD) (ours) and their partitions (# images).

features	with finetune				no finetune
	regression		classification		
layer	generic	specific	generic	specific	
pool1	0.27 / 7.67	0.26 / 7.68	0.28 / 7.59	0.28 / 7.62	0.28 / 7.61
pool2	0.35 / 6.70	0.28 / 6.97	0.29 / 7.09	0.29 / 7.06	0.28 / 7.09
pool3	**0.68 / 4.95**	**0.67 / 4.98**	0.67 / 5.00	0.67 / 5.00	0.67 / 5.00
pool4	0.66 / 5.04	0.67 / 5.01	**0.68 / 4.96**	**0.68 / 4.96**	**0.68 / 4.96**
pool5	0.63 / 5.28	0.63 / 5.18	0.66 / 5.14	0.66 / 5.14	0.66 / 5.14
fc6	0.56 / 5.72	0.55 / 5.72	0.55 / 5.81	0.55 / 5.81	0.55 / 5.81
fc7	0.04 / 7.46	0.00 / 7.91	0.45 / 6.28	0.45 / 6.29	0.45 / 6.28
pool3 + 4	0.42 / 6.33	0.34 / 6.81	0.43 / 6.27	0.43 / 6.30	0.44 / 6.29
fc6 + 7	0.53 / 5.88	0.51 / 5.95	0.55 / 5.74	0.55 / 5.75	0.55 / 5.74

Table 2. Temperature prediction average (R^2 / $RMSE$) results for deep features on Glasner *et al*. [3] (a–j) sequences. The best results for each setup are with bold.

2. Temperature prediction in outdoor scenes

The first part of this work is a study of deep features for temperature prediction in outdoor scenes and a direct comparison with the prior work of Glasner *et al*. [3] under the same benchmark.

2.1. Dataset and evaluation protocol

2.1.1 Glasner Dataset (GD)

For temperature prediction Glasner *et al*. [3] selected ten stable webcam sequences from the AMOS database, spanning two years. The webcams were taken from all over the USA. From these, only one image at 11AM, local time, per day and camera was kept. This roughly gives a number of images equal with the number of days in each sequence (see Table 1). The first year was used for training and the second year for testing. For temperature labels Glasner *et al*. used the data of nearby weather stations. The authors also preprocess the data to align the frames. Some sample images from each webcam are shown in Fig. 2.

2.1.2 Evaluation protocol

Glasner *et al*. [3] propose two measures for quantitative evaluation of the temperature prediction: the coefficient of determination (R^2):

$$R^2(y, \hat{y}) = 1 - \frac{\sum_{i=1}^{N}(\hat{y}_i - y_i)^2}{\sum_{i=1}^{N}(\bar{y}_i - y_i)^2} \quad (1)$$

and the root mean square error (RMSE):

$$RMSE(y, \hat{y}) = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(\hat{y}_i - y_i)^2}, \quad (2)$$

where y is the true response, $\hat{y}$ is the predicted response, $\bar{y} = \frac{\sum_{i=1}^{N} y_i}{N}$ is the average response. N is the number of samples. Note that Glasner *et al*. report 0 when R^2 takes negative values and so we do.

2.2. Prior Methods and Features

In their work [3], Glasner *et al*. report their results for temperature prediction using several settings (methods and features). The temperature from the previous year on the same day (LY) and the nearest image (NN Image) are taken as baselines. Other settings used include:

- Local Regression (LR), in which the prediction is a weighted combination of pixel individually used as a predictor in a least squares estimate of the temperature.
- Local Regression with a Temporal Window (LRTW) is a modified version of the above, in which the pixel intensities of the last nine days are also included.
- Global Regularized Regression (GRR) adds a sparsity constraint to Local Regression.
- Convolutional features (CNN) taken from the first fully connected layer of the VGG16 architecture pretrained for ImageNet image classification [15] are used as inputs to an SVM.
- Transient Image Attributes (TA) of Laffont *et al*. [9] are also used to fit an RBF-SVM.

2.3. Deep Features for Temperature Prediction

In this section we describe an experimental setup to test our hypothesis that deep features are effective for the task of temperature prediction.

First, we create three experimental setups to examine the effect of finetuning a convolutional neural network on the temperature data. As our pre-trained network, we select VGG-16, as this is the same network used by Glasner *et al*. in their work [3] and a top performing method for ImageNet image classification benchmark [15]. In this way our results are directly comparable to those of Glasner *et al*.

Our three setups are: 1) finetuning VGG-16 to directly predict temperature by regressing to a temperature value (only one output neuron); 2) finetuning to predict a temperature class label; and 3) no finetuning, and relying on the pretrained VGG-16 on ImageNet classification task. We used an Euclidean Loss function in the regression case and a multinomial logistic loss for classification.

Within each of these architectures, the network was finetuned either with training images from only one webcam sequence (specific finetuning) or with training images from all ten webcams (generic finetuning). 80% of the training images were used for training/finetuning, and the remaining

	(a)	(b)	(c)	(d)	(e)	(f)	(g)	(h)	(i)	(j)	(average)
LY	0.42 / 9.14	0.56 / 8.16	0.54 / 7.53	0.41 / 5.44	0.61 / 7.35	0.00 / 4.30	0.67 / 6.20	0.59 / 6.77	0.00 / 4.84	0.61 / 7.64	0.44 / 6.74
NN Image	0.47 / 8.72	0.59 / 7.83	0.51 / 7.73	0.15 / 6.51	0.13 / 10.92	0.00 / 4.57	0.16 / 9.89	0.70 / 5.83	0.00 / 4.44	0.62 / 7.47	0.33 / 7.39
LR	**0.67 / 6.85**	0.65 / 7.24	0.70 / 6.03	0.59 / 4.53	0.76 / 5.77	0.38 / 3.19	0.50 / 7.63	0.77 / 5.09	0.10 / 3.68	0.59 / 7.77	0.57 / 5.78
LRTW	0.61 / 7.52	0.69 / 6.86	0.72 / 5.82	**0.64 / 4.23**	0.79 / 5.39	**0.53 / 2.77**	0.54 / 7.35	0.76 / 5.22	0.11 / 3.67	0.58 / 7.85	0.60 / 5.67
GRR	0.00 / 18.16	0.78 / 5.74	0.00 / 35.02	0.00 / 11.37	0.00 / 43.51	0.10 / 3.84	**0.74 / 5.54**	0.00 / 13.86	0.23 / 3.41	0.46 / 8.91	0.23 / 14.94
CNN	0.49 / 8.55	0.79 / 5.59	0.71 / 5.96	0.24 / 6.17	0.61 / 7.36	0.48 / 2.90	0.39 / 8.48	0.79 / 4.88	0.43 / 2.93	0.66 / 7.12	0.56 / 5.99
TA	0.36 / 9.60	0.70 / 6.69	0.58 / 7.20	0.55 / 4.75	0.68 / 6.62	0.21 / 3.59	0.58 / 7.03	0.65 / 6.31	0.16 / 3.56	0.67 / 7.00	0.51 / 6.23
fc6	0.52 / 8.28	0.80 / 5.46	0.61 / 6.89	0.56 / 4.72	0.80 / 5.30	0.21 / 3.60	0.54 / 7.34	0.79 / 4.90	0.06 / 3.78	0.59 / 7.80	0.55 / 5.81
pool4	0.58 / 7.79	**0.84 / 4.87**	**0.79 / 5.03**	0.60 / 4.45	**0.87 / 4.22**	0.40 / 3.14	0.63 / 6.61	**0.80 / 4.72**	**0.52 / 2.70**	**0.76 / 6.01**	**0.68 / 4.96**

Table 3. Temperature prediction (R^2 / $RMSE$) results on Glasner *et al.*' GD dataset [3].

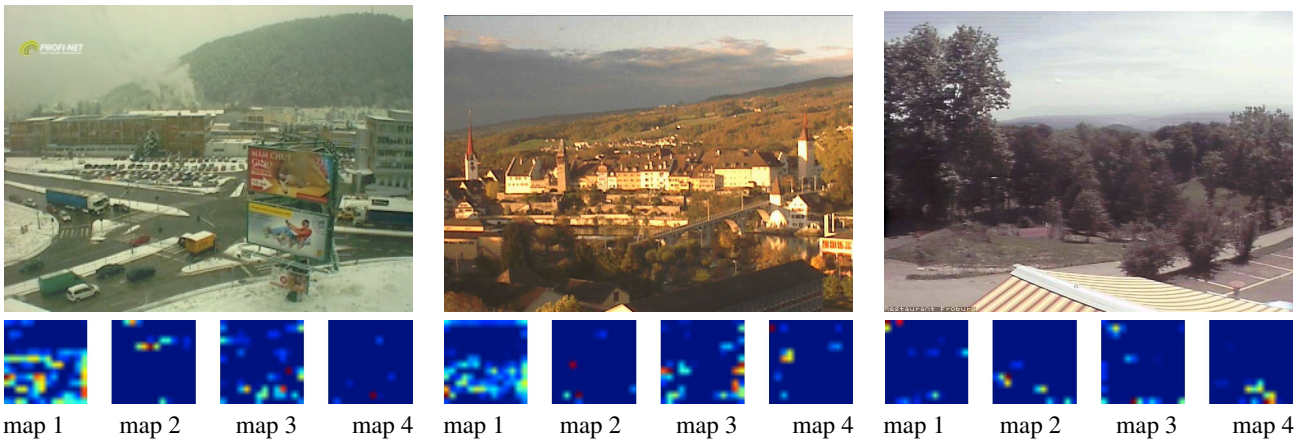

Figure 3. Examples of images and their corresponding first 4 maps within the pool4 layer of CNN.

20% for validation. We use Caffe [8] framework for all our experiments and finetuning. The learning rate is set to 1e-9, the update policy to *step*. Gamma is set to 0.1, momentum to 0.9 and weight decay to 5e-4.

Finetuning is stopped when the loss on the validation data reaches its lowest. For the regression setup, all finetuning is done with 10000 iterations. The number of iterations for each webcam in the classification setting for specific finetuning is as follows: (a) 2000, (b) 0, (c) 6000, (d) 6000, (e) 2000, (f) 100000, (g) 16000, (h) 10000, (i) 100000, (j) 0. For the generic case, we use 100000 iterations. Due to the relatively small number of training samples the finetuning did not improve the loss on the validation data for a couple of sequences.

As deep features, we extract neural activations from the pooling layers pool_1 – pool_5 and fully connected layers fc_6 – fc_8 layers. These features, as well as the combination of pool_3 with pool_4 and fc_6 with fc_7 are used to train support vector machines, with the same parameters as Glasner *et al.* [3], which are a linear ν-SVM with ν set to 0.5, and C to one. We use the LIBSVM [1] implementation from scikit-learn [14].

The results of our three setups using deep features on Glasner *et al.* [3] GD dataset for temperature prediction are reported in Table 2.

2.4. Findings

The results for temperature prediction (see Table 2) are overall good, and show a peak R^2 value of 0.68 and an error of $4.95°$ when using pool4 features.

Generic finetuning performs slightly better than specific finetuning, perhaps due to a larger amount of training data. More interestingly, finetuning does not improve significantly the accuracy over VGG16, and in fact at times decreases it. For fc6 and pool2 features we can benefit from finetuning. Overall, there was not enough data in our dataset to successfully finetune an image-classification network into a temperature prediction one.

Prediction first increases with layer depth and then decreases, peaking at the pool_3 or pool_4 layer in all setups. Intuitively, layers higher in the network encode semantic information about object category. The lower layers still retain a lot of spatial information, and respond to simpler attributes of the image such as color, gradient and shape. The attributes of an image relevant for predicting temperature are not semantic. Humans can roughly estimate the temperature outside regardless of whether a particular object is present or absent. While the presence of cars or parasols may provide some information, the saturation of the colors in the scene, the brightness of reflections from surfaces and the color of the sky are much stronger indicators for temperature.

We also find that fc_6 performs the best out of the fully connected layers, probably because it contains more spatial information than fc_7.

Interestingly, the concatenation of pool_3 and pool_4 performs much worse than either separately. This could be due to the fact that the dimensionality of this feature becomes

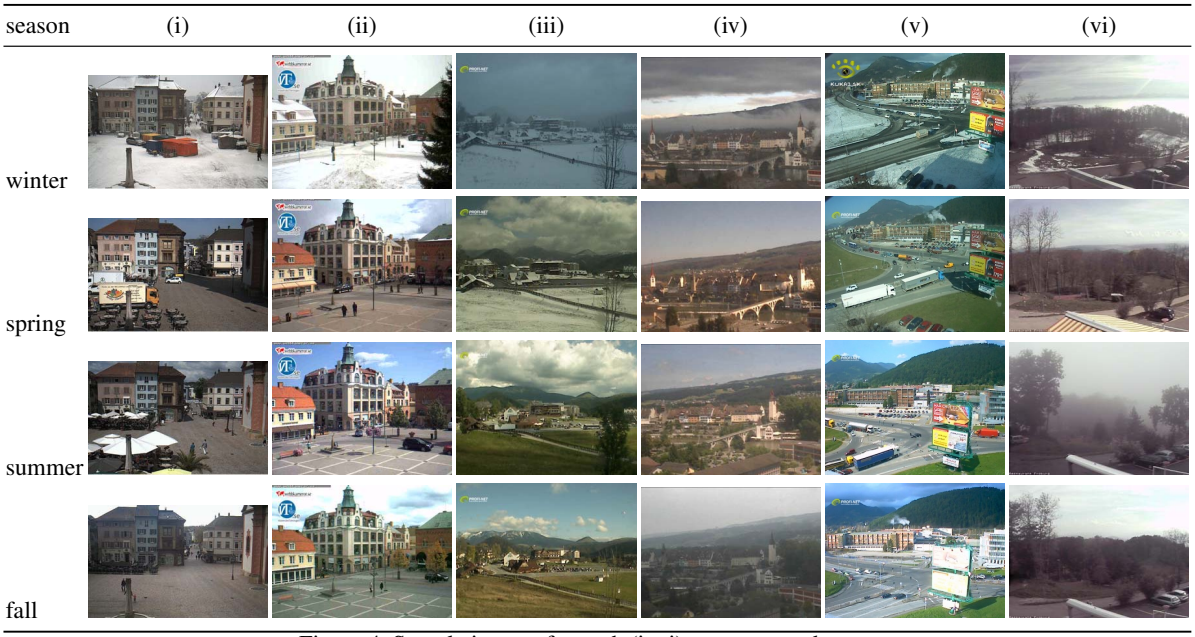

Figure 4. Sample images for each (i–vi) sequence and season.

too large: $200704 + 100352 = 301056$ dimensions. The combination of fc_6 and fc_7 is much smaller and has a performance in between what the two features achieve individually.

In Table 3 we show our results using deep features from $pool_4$ and fc_6 layers in comparison with the methods used by Glasner *et al*. and briefly described in section 2.2. We see that our methods outperform Glasner's methods on average and if taking the average over the best method in each case.

In their paper, Glasner *et al*. also report using the fc_6 feature extracted from VGG-16 without finetuning. The difference between their and our implementations is small, just 0.01 in R^2 and 0.18° Celsius.

Our method using features extracted from the $pool_4$ layer significantly improves on Glasner *et al*.'s previous best result by 0.08 in R^2 and $0.7°C$ in RMSE.

3. Time Prediction in Outdoor Scenes

Having determined that convolutional features are effective for the temperature prediction task, we attempt to use them on a related task – the time of the year prediction in outdoor scenes.

days												
weeks												
months	Jan	Feb	Mar	Apr	May	Jun	Jul	Aug	Sep	Oct	Nov	Dec
seasons	winter		spring			summer			fall			winter

Table 4. Diagram of how the year was partitioned into labels

hour											
time of day	morning			noon			afternoon			evening	

Table 5. Diagram of how the day was partitioned into labels

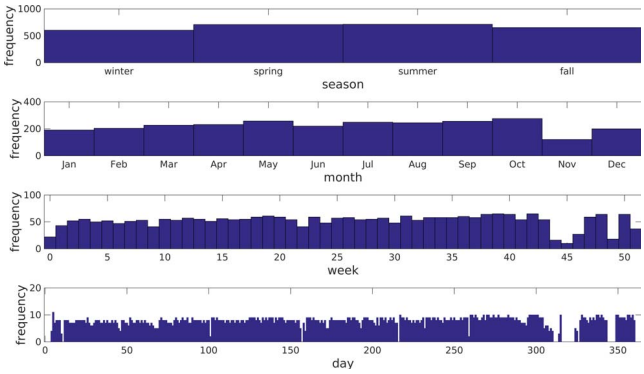

Figure 5. Distribution of the testing labels in TYD sequence (ii).

3.1. Dataset and evaluation protocol

3.1.1 Time of the Year Dataset (TYD)

For our second task of time of year prediction in outdoor scene images, we selected 6 image sequences spanning a minimum of two years. These images are not preprocessed or aligned, though most images are roughly aligned as the viewpoint of the camera does not change significantly over the years. One year of the two is taken as training data and the rest as testing. In total there are 23,198 images, 16,975 for training and 6,223 for testing. Some example images from the six sequences are shown in Fig 4.

We had images taken over a period of minimum two years, sampled periodically throughout the day. From these images we kept only images with daylight, mostly from 9:00 AM until 5:00 PM. Images with snow or fog obstructing the view completely were discarded. Following the example of Glasner *et al*., we take one year for training the

data and another for testing.

We discretized the time of year into time classes at the season, month, week, and day level. To compute the season, we use the formula

$$season = \left\lfloor \frac{(day + 31) \mod 366}{91.5} \right\rfloor \quad (3)$$

This means mostly that December, January and February are labeled winter, March, April and May as spring; June, July and August as summer, and September, October and November as fall. Months are determined in accordance with calendar months. Weeks were counted in seven-day intervals from the first day of the year. Table 4 shows how the year was divided into labels. Figure 5 shows the distribution of time labels in sequence (ii), which is almost uniform.

3.1.2 Evaluation protocol

The metrics used in the time classification task are the standard classification accuracy (%)

$$accuracy(y, \hat{y}) = 100 \frac{\sum_{y=\hat{y}} \mathbb{1}}{N} \quad (4)$$

and a modified RMSE, showing the average distance of the prediction from the true class. Since the time of year is cyclical, we compute this quantity in the following way:

$$RMSE(y, \hat{y}) = \sqrt{\frac{1}{N} \sum_{1}^{N} (\min(|\hat{y} - y|, D - (|\hat{y} - y|)))^2} \quad (5)$$

where N is the number of samples in the test set and D is the number of divisions in a year at a particular (season, month, week, day) level.

3.2. Deep features for time prediction

Predicting time of year is difficult, because there is so much variability within a season or a month, as shown in Fig. 6. The appearance of a scene changes most in winter, when snow falls. However, in Fig. 6 we see that snow is also present in some days of fall and spring, including in the middle of those seasons. Furthermore, days with cloudy skies in summer and in spring are more visually similar between themselves than two spring days with cloudy and clear skies.

To visualise the similarity and dissimilarity between images of sequence (ii) we display them in a 2-D grid using a projection of the fc_6 features with t-SNE [17], shown in Fig. 8. Fig. 8 shows winter as being the class or cluster that is most separated from the others, whereas spring, fall and summer overlap significantly. Although there is structure in the t-SNE embedding, it is not obvious from looking at the images that the images are divided into seasons.

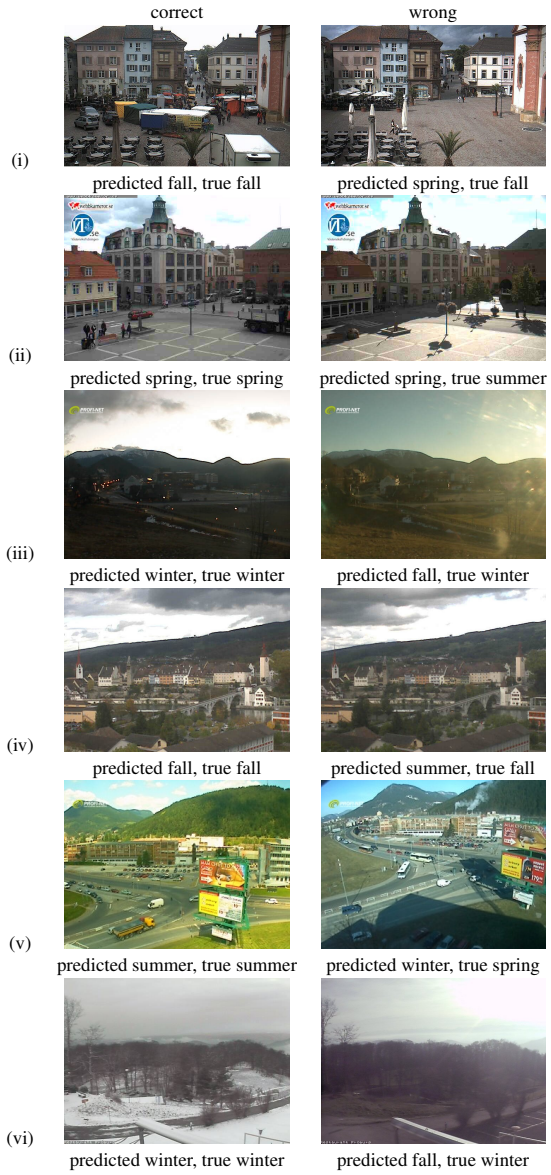

Table 6. Examples of correct and wrong season predictions.

To determine quantitatively how well convolutional features work for predicting time of year, two classifiers were used: nearest neighbor (1-NN) and a linear support vector machine (SVM). The parameters used for the SVM are a linear kernel with C set to 1. These were determined by cross-validation.

3.3. Findings

Table 8 shows the result for nearest neighbor classification and Table 7 shows the results using a linear SVM. The linear SVM yields the best results. The $pool_4$ layer outperforms the fc_6 layer in most of the cases. Classification accuracy improves from day to season level. Since we have defined a season to be 91.5 days long, we see that actually in predicting season, we have an average error of 0.67×91.5 days = 61.3 days, compared to prediction at the day level

Figure 6. Examples of images from sequence (iii) showing how the weather changes a lot during one season and along the year. The images are shown in chronological order.

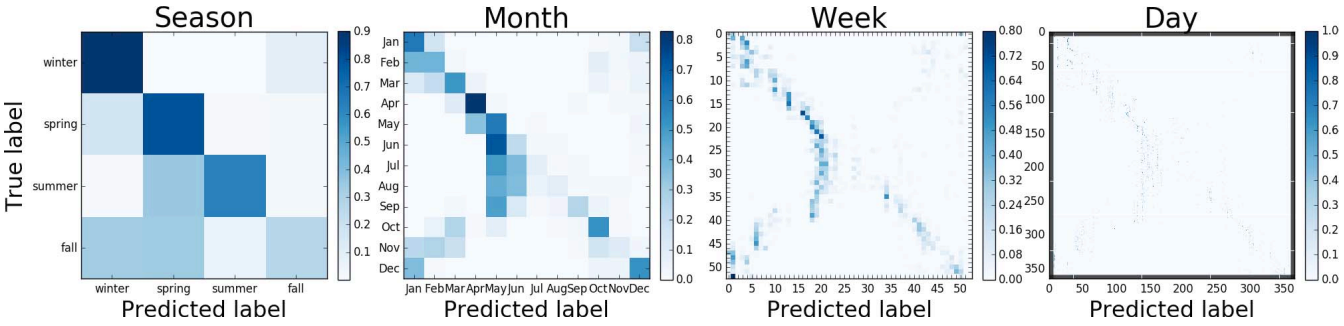

Figure 7. Average confusion matrices for time of the year prediction on the 6 sequences. The confusion happens mostly for neighboring days, weeks, months, or seasons. Best seen on screen.

time label	layer	(i)	(ii)	(iii)	(iv)	(v)	(vi)	(average)
season	pool4	76.9 / 0.52	67.1 / 0.63	92.3 / 0.46	42.1 / 0.88	64.0 / 0.59	69.4 / 0.92	**68.6 / 0.67**
	fc6	61.5 / 0.77	61.7 / 0.61	90.2 / 0.52	48.7 / 0.78	53.5 / 0.67	51.6 / 1.12	61.2 / 0.75
month	pool4	54.8 / 1.88	36.3 / 2.63	77.0 / 1.65	32.9 / 3.21	32.6 / 2.84	67.7 / 2.17	**50.2 / 2.40**
	fc6	38.9 / 2.62	29.8 / 2.50	73.6 / 1.82	19.7 / 3.51	20.9 / 2.98	44.8 / 2.21	38.0 / 2.61
week	pool4	21.2 / 9.84	9.6 / 11.46	6.4 / 7.71	7.9 / 14.35	10.5 / 12.61	2.3 / 9.05	**9.6 / 10.84**
	fc6	18.8 / 10.62	7.7 / 11.15	5.1 / 7.66	10.5 / 13.28	8.1 / 12.29	2.9 / 9.16	8.9 / 10.69
day	pool4	0.5 / 80.75	1.4 / 79.45	0.0 / 63.63	0.0 / 88.53	1.2 / 82.41	1.9 / 64.85	0.8 / **76.60**
	fc6	1.9 / 87.36	1.2 / 80.69	0.0 / 59.79	2.6 / 86.93	0.0 / 82.10	2.3 / 64.11	**1.3** / 76.83

Table 7. Comparison of SVM results in terms of classification accuracy (%) and RMSE error when using pool4 or fc6 deep features.

time label	layer	(i)	(ii)	(iii)	(iv)	(v)	(vi)	(average)
season	pool4	58.7 / 0.80	64.8 / 0.60	85.5 / 0.63	59.2 / 0.62	47.7 / 0.84	62.3 / 1.02	**63.0 / 0.75**
	fc6	48.1 / 0.87	57.4 / 0.69	72.8 / 0.87	63.2 / 0.60	43.0 / 0.85	44.2 / 1.25	54.8 / 0.85
month	pool4	43.8 / 2.65	34.8 / 2.56	69.4 / 1.85	36.8 / 2.84	29.1 / 2.63	53.9 / 2.11	**44.6 / 2.44**
	fc6	36.1 / 2.62	25.4 / 2.68	53.2 / 2.29	36.8 / 3.07	26.7 / 2.54	38.1 / 2.09	36.0 / 2.55
week	pool4	15.9 / 11.49	10.4 / 11.15	1.7 / 8.13	3.9 / 12.55	14.0 / 11.62	3.9 / 9.33	8.3 / **10.71**
	fc6	15.4 / 11.41	8.1 / 11.69	2.6 / 10.13	7.9 / 13.52	12.8 / 11.23	4.2 / 9.43	**8.5** / 11.23
day	pool4	1.0 / 80.51	1.3 / 78.24	0.0 / 57.02	1.3 / 87.88	0.0 / 81.30	1.9 / 65.36	0.9 / **75.05**
	fc6	1.4 / 79.84	1.0 / 82.01	0.0 / 71.31	1.3 / 94.75	1.2 / 78.61	2.3 / 65.89	**1.2** / 78.73

Table 8. Comparison of 1-NN results in terms of classification accuracy (%) and RMSE error when using pool4 or fc6 deep features.

where the average error is 77.55 days.

As discussed above, time prediction is a challenging task because days of different time labels (season, month, week) look very alike. Fig. 7 presents confusion matrices. We can

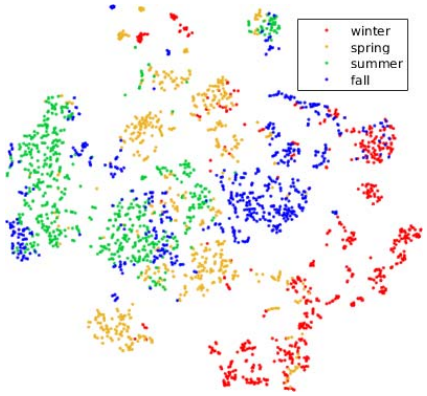

Figure 8. t-SNE embedding of testing images from webcam (ii)

see that, indeed, winter is the season with the highest classification accuracy. Also, we notice a strong bias in prediction to spring.

Despite the difficulty of the task, convolutional features perform well in the task of predicting time, 68.6% is the achieved accuracy at season level and 50.2% at month level.

To better understand what the deep features represent, we visualise the first four responses of the pool_4 layer in Fig 3. In the first two images, map 1 responds to the man-made structures in the bottom half of the scene. It is difficult to interpret the other maps.

Finally, in Fig. 6 we display some cases in which the season was and was not correctly predicted. In the left column, the dominant color of the image coincides with our intuition about the season. For example the saturated yellow in sequence (v) strongly suggests summer, while the gray in (iv) suggests fall. Many of the images in the right column could have plausibly belonged to the predicted season.

3.4. Influence of the Training size

We claim that our attempts at finetuning VGG-16 in sec. 2.4 were not successful due to having too little training examples (3068 in the generic scenario). To validate this claim, we now attempt to finetune VGG-16 using more images for the time prediction task. We select sequence (a), AMOS id 17603, from Glasner *et al.*'s dataset and download all available images from the AMOS online database, yielding 28427 images spanning 4.4 years after downsampling to a uniform sampling rate. Year 1 (5623 images) and year 2 (7111) were used for training, year 3 (6353) for validation and the remaining year 4 and part of 5 (9340) for testing. We exclude nighttime pictures and keep only images taken between 13:00 and 23:00 UTC (approx. 8:00 to 18:00 in local time). To see the effect of increasing the amount of training data, we finetuned VGG-16 first with only images from year 1 and then with images from years 1 and 2 for predicting the season, month, week and day as before. Table 9 shows the (cyclical) RMSE for each of the experiments. We see that increasing the amount of training data decreases the loss in all cases. As a reference, we include the average RMSE values from time prediction using pool_4 features on our TYD dataset, and see that we achieve better performance in finetuning the net directly for the task.

	season	month	week	day	time of day	hour
train year 1	0.84	2.85	12.09	82.67	35.31/0.68	20.50/1.79
train year 1+2	0.59	2.15	9.60	73.07	79.64/0.37	59.24/0.83
TYD $_{(\text{SVM, pool}_4)}$	0.67	2.40	10.84	76.60		

Table 9. CNN (VGG-16) prediction results on sequence (a) when finetuning with one and two years of train data. We report RMSE and accuracy (%)/RMSE for time of the day and hour prediction.

Train data	year 1	year 1+2	year 1	year 1+2
Finetune	×	×	✓	✓
pool_4, SVM	37.36/0.70	37.14/0.64	38.43/0.70	37.00/0.64
fc_6, SVM	41.84/0.83	36.08/0.65	44.50/0.84	35.79/0.62
CNN (direct)			/0.84	/0.59

Table 10. Season prediction results (accuracy % / RMSE) on (a).

In Table 10, we compare SVM predictions of the season using finetuned features from the neural network to using features without finetuning and find that increasing the number of training examples increases performance: in both the scenarios with and without finetuning, using two years of training data decreases the loss. Here, the best loss is achieved by the direct CNN with finetune on 2 years.

Time of the day prediction. Furthermore, using the same dataset (a), we also attempt to predict the time of day (morning, noon, afternoon and evening) and the hour of the day (partitions shown in Table 3) and report both the accuracy and RMSE in Table 9. An additional year of training data improves the time of day prediction accuracy from 35% to 80%, and the hour prediction from 20% to 60%, which is very impressive given that the appearance of a scene changes depending not only on the hour of the day, but also on the day of the year.

4. Conclusions

In this paper, we studied the effectiveness of deep features for ambient temperature and time of the year prediction in outdoor scenes. Our main findings are that the pooling layers provide better features than the commonly employed fully connected layers for these tasks and that the CNN finetuning on training outdoor data leads to small improvements over the CNN pretrained for image classification on a much larger dataset (ImageNet). Our results are $0.7°C$ better than those of Glasner *et al.* [3] using handcrafted features on temperature prediction, showing the power of the deep features. On our proposed dataset and time of the year prediction task we obtain robust performance (especially at month and season level). To the best of our knowledge, it is the first attempt at predicting the time of the year and time of day in outdoor scenes. The good accuracies achieved on diverse scenes are promising and show that we have the tools to turn cameras into crude but effective ambient temperature and time sensors.

Acknowledgments. This work was supported by The ETH General Fund (OK) and the ERC project VarCity (#273940). We thank NVIDIA for donating a Tesla K40 GPU.

References

[1] C.-C. Chang and C.-J. Lin. LIBSVM: A library for support vector machines. *ACM Transactions on Intelligent Systems and Technology*, 2:27:1–27:27, 2011. Software available at http://www.csie.ntu.edu.tw/~cjlin/libsvm. 4

[2] M. Elhoseiny, S. Huang, and A. Elgammal. Weather classification with deep convolutional neural networks. In *Image Processing (ICIP), 2015 IEEE International Conference on*, pages 3349–3353, Sept 2015. 2

[3] D. Glasner, P. Fua, T. Zickler, and L. Zelnik-Manor. Hot or not: Exploring correlations between appearance and temperature. In *The IEEE International Conference on Computer Vision (ICCV)*, December 2015. 1, 2, 3, 4, 8

[4] M. Islam, N. Jacobs, H. Wu, and R. Souvenir. Images+ weather: Collection, validation, and refinement. In *IEEE Conference on Computer Vision and Pattern Recognition Workshop on Ground Truth*, volume 6, 2013. 2

[5] N. Jacobs, W. Burgin, N. Fridrich, A. Abrams, K. Miskell, B. H. Braswell, A. D. Richardson, and R. Pless. The global network of outdoor webcams: Properties and applications. In *Proceedings of the 17th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems*, GIS '09, pages 111–120, New York, NY, USA, 2009. ACM. 2

[6] N. Jacobs, W. Burgin, R. Speyer, D. Ross, and R. Pless. Adventures in archiving and using three years of webcam images. In *Computer Vision and Pattern Recognition Workshops, 2009. CVPR Workshops 2009. IEEE Computer Society Conference on*, pages 39–46, June 2009. 2

[7] N. Jacobs, N. Roman, and R. Pless. Consistent temporal variations in many outdoor scenes. In *Computer Vision and Pattern Recognition, 2007. CVPR '07. IEEE Conference on*, pages 1–6, June 2007. 2

[8] Y. Jia, E. Shelhamer, J. Donahue, S. Karayev, J. Long, R. Girshick, S. Guadarrama, and T. Darrell. Caffe: Convolutional architecture for fast feature embedding. *arXiv preprint arXiv:1408.5093*, 2014. 4

[9] P.-Y. Laffont, Z. Ren, X. Tao, C. Qian, and J. Hays. Transient attributes for high-level understanding and editing of outdoor scenes. *ACM Trans. Graph.*, 33(4):149:1–149:11, July 2014. 2, 3

[10] C. Lu, D. Lin, J. Jia, and C.-K. Tang. Two-class weather classification. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2014. 2

[11] C. Murdock, N. Jacobs, and R. Pless. Building dynamic cloud maps from the ground up. In *The IEEE International Conference on Computer Vision (ICCV)*, December 2015. 2

[12] S. G. Narasimhan and S. K. Nayar. Vision and the atmosphere. *International Journal of Computer Vision*, 48(3):233–254, 2002. 2

[13] S. G. Narasimhan, C. Wang, and S. K. Nayar. *Computer Vision — ECCV 2002: 7th European Conference on Computer Vision Copenhagen, Denmark, May 28–31, 2002 Proceedings, Part III*, chapter All the Images of an Outdoor Scene, pages 148–162. Springer Berlin Heidelberg, Berlin, Heidelberg, 2002. 2

[14] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011. 4

[15] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014. 2, 3

[16] M. Szummer and R. W. Picard. Indoor-outdoor image classification. In *Content-Based Access of Image and Video Database, 1998. Proceedings., 1998 IEEE International Workshop on*, pages 42–51, Jan 1998. 2

[17] L. Van der Maaten and G. Hinton. Visualizing data using t-sne. *Journal of Machine Learning Research*, 9(2579-2605):85, 2008. 6